Assignment 3

Question 1

Derive input space partitioning test inputs for the GenericStack class.
Following the steps presented in slides in chapter 6, using interface-Based Approach.

Two inputs: Object, state

Characteristics, blocks and values:

- Whether the stack is empty.

{ true (Value stack = [])
{ false (Values stack = [\cat"], [\cat", \hat"])

- The size of the stack.

{ 0 (Value stack = [])
{ 1 (Possible values stack = [\cat"], [null])
{ more than 1 (Possible values stack = [\cat", \hat"], [\cat", null], [\cat", \hat",
\ox"])

Whether the stack contains null entries
{ true (Possible values stack = [null], [null, \cat", null])
{ false (Possible values stack = [\cat", \hat"], [\cat", \hat", \ox"])

A typical characteristic for Object x is
_ Whether x is null.
{ true (Value x = null)
{ false (Possible values x = \cat", \hat", \")

There are also characteristics that involves the combination of Object x and the stack state. One is:
_ Does Object x appear in the stack?
{ true (Possible values: (null, [null, \cat", null]), (\cat", [\cat", \hat"]))
{ false (Possible values: (null, [\cat"]), (\cat", [\hat", \ox"]))

Question 2

Base Cases: (2, 2, 2) and (1, 2, 2)
With (2, 2, 2):
{(2, 2, 2), (0, 2, 2), (-1, 2, 2), (2, 1, 2), (2, 0, 2), (2, -1, 2), (2, 2, 1), (2, 2, 0), (2, 2, -1)}

With (1, 2, 2):
{(1, 2, 2), (0, 2, 2), (-1, 2, 2), (1, 1, 2), (1, 0, 2), (1, -1, 2), (1, 2, 1), (1, 2, 0), (1, 2, -1)}

Redundant Cases: (0, 2, 2) and (-1, 2, 2)

Total 16 Cases:
{(2, 2, 2), (0,2,2), (-1, 2, 2), (2, 1, 2), (2, 0, 2), (2, -1, 2), (2, 2, 1), (2, 2, 0), (2, 2, -1),
(1, 2, 2), (1, 1, 2), (1, 0, 2), (1, -1, 2), (1, 2, 1), (1, 2, 0), (1, 2, -1)}

Question 3

a) Yes Validity of S1 satisfies the completeness property as all values that can be inputted fits into one of the categories. It will either be null, {} or have at least one element.

b) Yes, Validity of S1 satisfies the disjointness property as no inputted value will fit into any two categories. Each value will fit in exactly one category.

c) No. S1 =(1,2,3) S2={1, 4, 5}

d) No Relation between s1 and s2 does not satisfy the completeness property because if s1 and s2 are both empty it can be classified as representing the same set or not having any elements in common (since no elements are in the sets).

Result: 1 + 2 + 3 = 6 test requirements total

Question 4

Covers 3 more invalid triangle types, Two test cases are from Question 2. I also noticed 0, 0, 0 was not added in the test file so I added that one too.

- When side 1 is 0 and sides 2 and 3 are equal (not a valid triangle)
- When side 1 is negative and sides 2 and 3 are equal (not a valid triangle)
- When all sides are 0 and the sum equals 0 (not a valid triangle)

Code Added:

```
list.add(new Object[] {0, 2, 2, Triangle.INVALID});
list.add(new Object[] {-1, 2, 2, Triangle.INVALID});
list.add(new Object[] {0, 0, 0, Triangle.INVALID});
```